**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Факультет компьютерных технологий и информатики**

**Кафедра систем автоматизированного проектирования**

отчет

**по производственной практике (преддипломной практике)**

Тема: **Разработка веб-приложения для ведения ведомостей текущего контроля**

|  |  |  |
| --- | --- | --- |
| Студент гр. 9302 |  | Давтян С. Д. |
| Руководитель практики  к.т.н., доцент |  | Михалков В. А. |
| Согласовано:  Руководитель ВКР  д.т.н., профессор |  | Герасимов И. В. |
| Консультант по ВКР  к.т.н. |  | Кузьмин С. А. |

Санкт-Петербург

2023

**ЗАДАНИЕ**

**на ПРЕДДИПЛОМНУЮ практику**

|  |
| --- |
| Студент: Давтян С. Д. |
| Группа: 9302 |
| Тема практики: Разработка веб-приложения для ведения ведомостей текущего контроля |
| Цели преддипломной практики:   1. Получение практических навыков, необходимых для выполнения выпускной квалификационной работы. 2. Сбор материала, необходимого для написания выпускной квалификационной работы.   Задание на преддипломную практику:   * + 1. Согласование технического задания на разработку приложения;     2. Обзор литературы по теме работы;     3. Проектирование архитектуры приложения;     4. Разработка информационной части приложения;     5. Разработка интерфейса клиентской части приложения;     6. Разработка кода клиентской части приложения;     7. Разработка кода серверной части приложения;     8. Тестирование приложения. |

|  |  |  |
| --- | --- | --- |
| Сроки прохождения практики: 04.04.2023 – 17.05.2023 | | |
| Дата сдачи отчета: 16.05.2023 | | |
| Дата защиты отчета: 17.05.2023 | | |
| Студент |  | Давтян С. Д. |
| Руководитель практики  к.т.н., доцент |  | Михалков В. А. |
| Согласовано:  Руководитель ВКР  д.т.н., профессор |  | Герасимов И. В. |
| Консультант по ВКР  к.т.н. |  | Кузьмин С. А. |

**ГРАФИК выполнения заданий**

**ПРЕДДИПЛОМНОЙ ПРАКТИКИ**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент: Давтян С. Д. | | | | | |
| Группа: 9302 | | | | | |
| Тема практики: Разработка веб-приложения для ведения ведомостей текущего контроля | | | | | |
| **№ п/п** | **Наименование работ** | | | **Срок выполнения** |
| 1 | Согласование технического задания на разработку приложения | | | 04.04 – 05.04 |
| 2 | Обзор литературы по теме работы | | | 06.04 – 13.04 |
| 3 | Проектирование архитектуры приложения | | | 14.04 – 17.04 |
| 4 | Разработка информационной части приложения | | | 18.04 – 22.04 |
| 5 | Разработка интерфейса клиентской части приложения | | | 23.04 – 27.04 |
| 6 | Разработка кода клиентской части приложения | | | 28.04 – 01.05 |
| 7 | Разработка кода серверной части приложения | | | 02.05 – 09.05 |
| 8 | Тестирование приложения | | | 10.05 – 11.05 |
| 9 | Оформление отчета по преддипломной практике | | | 12.05 – 16.05 |
| 10 | Сдача отчета и защита итогов практики | | | 17.05 |
| Студент | |  | Давтян С. Д. | | |
| Руководитель практики  к.т.н., доцент | |  | Михалков В. А. | | |

АННОТАЦИЯ

Целью данной выпускной квалификационной работы является создание веб-приложения для ведения ведомостей текущего контроля, которое может быть использовано для организации дистанционного взаимодействия преподавателя со студентами по выполняемым ими работам в течение семестра.

В данном приложении пользователям сначала нужно зарегистрироваться или авторизоваться. В зависимости от роли пользователя будут доступен дальнейший функционал. Если пользователь является студентом, то он сможет просматривать задания, высылать свои работы, просматривать оценки и комментарии к ним. Если пользователь является преподавателем, то он сможет создавать ведомости, вносить в нее студентов, количество работ, скачивать файлы, отправленные студентами, а также проставлять оценки и писать свои замечания.

Основное отличие разрабатываемого приложения от его аналогов состоит в возможности отобразить в одной ячейке таблицы (ведомости) как отметку (или оценку) за выполненную работу, так и историю замечаний и прикреплённых файлов по ней.

Для реализации задуманного были использованы следующие технологии:

* Клиентская часть приложения разрабатывалась с помощью фреймворка Bootstrap5.
* Серверная часть приложения разрабатывалась с помощью языка программирования Python при активном использовании библиотеки Flask.
* Для работы с файлами на стороне сервера использовалась …
* Для взаимодействия с СУБД на стороне сервера используется библиотека SQLAlchemy.

ABSTRACT

The purpose of this graduate qualification work is to create a web application to keep records of current control, which can be used to organize remote interaction between the teacher and the students on the work they perform during the semester.

In this application, users must first register or log in. Depending on the user's role, further functionality will be available. If the user is a student, they will be able to view assignments, send their work, and view grades. If the user is an instructor, they will be able to create a worksheet, enter students, number of papers, receive files sent by students, and write their notes and assign grades.

The main difference between the application under development and its analogues is the ability to display both the mark (or grade) for the completed work and the history of comments and attached files on it in a single table cell (sheet).

The following technologies were used to implement the idea:

* The client part of the application was developed using the Bootstrap5 framework.
* The server part of the application was developed using the Python programming language with active use of the Flask library.
* The SQLAlchemy library is used to interact with the DBMS on the server side.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 9](#_Toc135080506)

[1. ПОСТАНОВКА ЗАДАЧИ 10](#_Toc135080507)

[1.1. Тема проекта 10](#_Toc135080508)

[1.2. Главные задачи 10](#_Toc135080509)

[1.3. Детальное описание приложения 10](#_Toc135080510)

[1.4. Описание предметной области 11](#_Toc135080511)

[1.5. Функционал приложения 12](#_Toc135080512)

[1.6. Используемые технологии 13](#_Toc135080513)

[1.7. Этапы разработки приложения 14](#_Toc135080514)

[2. ВЫБОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ 15](#_Toc135080515)

[2.1. Выбор языка программирования 15](#_Toc135080516)

[2.2. Выбор технологий 24](#_Toc135080517)

[2.3. Выбор базы данных 26](#_Toc135080518)

[2.4. Выбор среды разработки 27](#_Toc135080519)

[3. РАЗРАБОТКА ПРИЛОЖЕНИЯ 31](#_Toc135080520)

[3.1. Архитектурная часть приложения 31](#_Toc135080521)

[3.2. Информационная часть приложения 31](#_Toc135080522)

[3.3. Клиентская часть приложения 33](#_Toc135080523)

[3.4. Серверная часть приложения 45](#_Toc135080524)

[3.5. Тестирование приложения 45](#_Toc135080525)

[ЗАКЛЮЧЕНИЕ 47](#_Toc135080526)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 48](#_Toc135080527)

ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

СУБД – система управления базами данных – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием [баз данных](https://ru.wikipedia.org/wiki/%D0%91%D0%B0%D0%B7%D0%B0_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85).

ВВЕДЕНИЕ

Актуальность разработки приложений для организации учебного процесса непрерывно возрастает. Многие учебные заведения используют дистанционное взаимодействие студентов с преподавателем. Одной из проблем, которая возникает при таком переходе, является необходимость ведения ведомости текущего контроля. Традиционные методы ведения ведомостей на бумажных носителях уже не всегда эффективны и удобны в использовании. Разработка данного веб-приложения позволит существенно облегчить этот процесс.

Существенным отличием данной работы от уже существующих аналогов состоит в возможности отобразить в одной ячейке таблицы (ведомости) как отметку (или оценку) за выполненную работу, так и историю замечаний и прикреплённых файлов по ней.

Также текущая разработка может быть полезна не только в области образования, но и в других сферах: например, в медицине для ведения медицинских документов или в промышленности для контроля качества продукции.

# ПОСТАНОВКА ЗАДАЧИ

## Тема проекта

Разработка веб-приложения для ведения ведомостей текущего контроля.

## Главные задачи

1. Организация дистанционного взаимодействия преподавателя со студентами по выполняемым ими работам в течение семестра.
2. Разработка приложения с веб-интерфейсом, в котором преподаватель может проставлять отметки о проверенных им работах (а также в комментариях оставлять замечания по ним), а студенты могут читать комментарии, а также добавлять файлы работ.

## Детальное описание приложения

Приложение является многопользовательской системой, в которой пользователи могут быть двух видов: преподаватели и студенты.

Преподаватели решают задачи:

* создание предметных дисциплин;
* создание ведомостей по дисциплинам;
* проверка присланных работ и написание по ним замечаний;
* проставление оценок в ведомость.

Студенты решают задачи:

* добавление файла решения задачи;
* просмотр замечаний от преподавателя;
* добавление комментариев к посылаемому файлу;
* добавление исправленного файла.

## Описание предметной области

Таблица «Пользователь»:

* id;
* имя;
* фамилия;
* отчество (если есть);
* e-mail;
* пароль;
* роль;
* id группы;
* список работ;

Таблица «Предмет»:

* id;
* название;
* ведомости;

Таблица «Группа»:

* id;
* номер группы;
* студенты;

Таблица «Ведомость»:

* id;
* учебный год;
* семестр;
* id предмета;
* группы;
* работы;

Таблица «Столбик с работами»:

* id;
* название;
* список работ;
* id ведомости;

Таблица «Работа»:

* id;
* название файла программы;
* оценка за программу;
* название файла отчета;
* оценка за отчет;
* id столбика с работами;
* id пользователя;
* комментарии;

Таблица «Комментарий»:

* id;
* текст комментария;
* время комментария;
* id работы;

## Функционал приложения

* Модуль «Преподаватель»:
  + регистрация нового преподавателя;
  + авторизация преподавателя;
  + редактирование данных профиля преподавателя;
  + удаление профиля преподавателя;
  + уточнение дисциплины и состава контрольных точек по ней (с автоматической генерацией ведомости);
  + просмотр файлов, полученных от студентов;
  + проставление отметки о выполнении работы;
  + запись комментария к работе с замечаниями по работе студента;
  + редактирование отметки о выполнении работы;
  + редактирование комментария к выполненной работе;
  + удаление отметок о выполнении работ;
  + удаление комментариев к выполненным работам;
  + добавление новой редакции файла к контрольной точке.
* Модуль «Студент»:
  + регистрация нового студента;
  + авторизация студента;
  + редактирование данных профиля студента;
  + удаление профиля студента;
  + добавление файла по выполненной работе;
  + просмотр отметки о выполнении работы;
  + просмотр комментария к работе студента;
  + добавление новой редакции файла к контрольной точке;
  + удаление файла работы.

## Используемые технологии

* На стороне клиента:

фреймворк Bootstrap5 […] (HTML5 [], CSS [], JavaScript []);

библиотека jQuery;

инструмент Jinja2.

* На стороне сервера:

язык программирования Python;

фреймворк Flask;

библиотека Os (для работы с файлами).

* На стороне базы данных:

СУБД SQLite;

библиотека SQLAlchemy.

* Среда разработки:

Visual Studio Code 11.76.2;

PyCharm Community Edition 2023.1.1.

## Этапы разработки приложения

1. Согласование ТЗ;
2. Разработка архитектуры приложения;
3. Разработка информационной части (базы данных) приложения;
4. Дизайн клиентской части приложения;
5. Разработка кода клиентской части приложения;
6. Разработка кода серверной части приложения;
7. Тестирование приложения.

# ВЫБОР ИСПОЛЬЗУЕМЫХ ТЕХНОЛОГИЙ

## Выбор языка программирования

Язык программирования является основным инструментом при написании любого программного обеспечения. От него зависит сложность написания тех или иных элементов, а также их производительность. Поэтому создание проекта начинается именно с выбора языка программирования.

При создании веб-приложения стоит учитывать, что нужно написать клиентскую и серверную части приложения.

**Клиентская часть.**

Основная задача клиентской части веб-приложения - облегчить работу конечному пользователю и предоставить ему удобный интерфейс для взаимодействия с серверной частью приложения. Для реализации этой задачи были выбраны языки программирования HTML, CSS и JavaScript.

HTML (HyperText Markup Language) - это язык разметки, который определяет структуру и содержание веб-страницы […]. Он обеспечивает базовую структуру документа, определяет заголовки, параграфы, списки, таблицы и многие другие элементы страницы.

CSS (Cascading Style Sheets) - это язык стилей, который используется для оформления и стилизации элементов HTML. Он позволяет управлять цветом, шрифтами, размерами и расположением элементов страницы.

JavaScript - это язык программирования, который используется для создания интерактивных элементов веб-страницы. Он позволяет создавать динамические элементы, такие как выпадающие меню, всплывающие окна, анимации и многие другие.

Все эти языки программирования взаимодействуют между собой, чтобы создать удобный и функциональный интерфейс для конечного пользователя. HTML формирует структуру страницы, CSS добавляет стили и оформление, а JavaScript обеспечивает интерактивность и динамичность приложения.

Благодаря этому сочетанию языков программирования, разработчики могут создавать профессиональные веб-приложения, которые удобны и функциональны для пользователей.

**Серверная часть.**

Для написания серверной части веб-приложения используются различные языки программирования. Каждый из этих языков имеет свои преимущества и недостатки, поэтому выбор зависит от конкретных требований к проекту и предпочтений разработчика.

Приведём некоторые допустимые для такой цели языки:

* + 1. Java,
    2. PHP,
    3. Python,
    4. Node.js (JavaScript),
    5. C#.

Коротко рассмотрим каждый из них в отдельности, оценивая плюсы и минусы:

* + 1. **Java**

Java - это мощный, независимый от платформы, объектно-ориентированный язык программирования, который был разработан в 1995 году. С тех пор Java стала очень популярным выбором для создания различных приложений, в том числе и серверных.

Одной из основных причин популярности Java для серверной разработки является ее многопоточная модель. Java обеспечивает надежную и безопасную среду выполнения кода, благодаря чему позволяет тщательно контролировать доступ к ресурсам и обеспечивать высокий уровень безопасности. Кроме того, многопоточность Java позволяет масштабировать приложения и обрабатывать большой объем запросов одновременно, что особенно важно для веб-приложений.

Одним из главных преимуществ Java является его кроссплатформенность - код, написанный на этом языке, может запускаться на любой системе, где поддерживается соответствующая виртуальная машина.

Java также имеет С-подобный синтаксис, который является негласным стандартом среди программистов. Это делает его легким для изучения и использования для разработчиков, знакомых с другими С-подобными языками.

В Java также есть множество библиотек и фреймворков для разработки веб-приложений, таких как Spring, Hibernate, Struts, JSF, которые упрощают процесс разработки и позволяют создавать мощные и гибкие серверные приложения.

Java также обладает широкой поддержкой в сообществе разработчиков и имеет множество ресурсов для обучения и поддержки.

Хотя Java – один из самых популярных языков для создания серверной части веб-приложений, он также имеет некоторые недостатки:

1. Низкая производительность: Java не является самым производительным языком программирования, поскольку использует виртуальную машину для выполнения кода, а также требует дополнительных ресурсов для запуска. Это может привести к задержкам при обработке большого количества запросов.

2. Высокое потребление памяти: Java использует много памяти для загрузки кода, а также для хранения объектов в оперативной памяти. Это может привести к проблемам с масштабированием приложения при росте количества пользователей.

3. Сложность кодирования: Java – довольно сложный язык программирования, требующий от разработчиков высокой квалификации и опыта. Это может привести к тому, что разработка серверной части веб-приложения на Java будет занимать больше времени и ресурсов.

Таким образом, Java является одним из лучших языков программирования для написания серверной части веб-приложений благодаря своей многопоточной модели, мощным инструментам разработки, высокой степени безопасности и поддержке сообщества разработчиков, но при этом имеет ряд недостатков, с которыми придется столкнуться при работе с ним.

* + 1. **PHP**

PHP - это интерпретируемый язык программирования, который используется для создания динамических веб-сайтов и веб-приложений. Он популярен и широко используется для создания серверной части веб-приложений.

PHP обладает рядом преимуществ, которые делают его подходящим для написания серверной части веб-приложения:

1. Простота в изучении: PHP имеет простой и понятный синтаксис, который легко понять любому программисту.

2. Бесплатность и открытый исходный код: PHP бесплатен, доступен для скачивания и имеет открытый исходный код. Это позволяет программистам разрабатывать приложения с нуля и делать их по своему усмотрению.

3. Широкий выбор фреймворков: PHP имеет множество фреймворков, которые помогают программистам разрабатывать приложения быстрее и более эффективно.

4. Большое сообщество: PHP имеет огромное сообщество разработчиков, которые создают различные библиотеки и плагины, что упрощает процесс разработки.

К недостаткам можно отнести:

1. Неэффективность работы с памятью: PHP не очень хорошо управляет памятью, что может привести к утечкам памяти и низкой производительности.

2. Неустойчивость: Некоторые версии PHP не стабильны и могут приводить к неожиданным ошибкам.

3. Небольшое количество инструментов для тестирования: Не так много инструментов для тестирования PHP-приложений, что может затруднить тестирование и отладку.

4. Некоторые проблемы безопасности: PHP-приложения могут стать легкой добычей для хакеров, так как в PHP присутствует множество уязвимостей.

5. Медленный процесс разработки: Разработка приложений на PHP может занять много времени и ресурсов, особенно если вы не знакомы с языком.

6. Большое количество старых кодов: Простота использования и распространения PHP для Веб-приложений привела к большому количеству устаревшего кода в Интернете, что может привести к проблемам совместимости при новой разработке.

7. Архитектурный дизайн: PHP имеет ограниченные возможности для реализации сложных архитектурных решений, что может привести к проблемам масштабируемости в случае успешной работы приложения и необходимости его масштабирования.

8. Ограниченные возможности модульности: PHP не поддерживает полноценную модульность, что может привести к конфликтам импорта и дублированию кода при разработке приложений.

Таким образом, PHP остается одним из наиболее используемых языков программирования для написания серверной части веб-приложений и продолжает быть востребованным в сфере веб-разработки, однако при работе с ним стоит учесть недостатки, которые могут сильно повлиять на разработку приложения.

* + 1. **Python**

Python - высокоуровневый язык программирования, который с приходом новых версий улучшается и становится все более удобным в использовании. Сами основатели языка описывают его как простой в изучении, но мощный в построении инструментов.

Python обладает множеством библиотек и фреймворков, которые можно использовать для написания серверной части веб-приложения. Некоторые из них, такие как Django и Flask, широко используются в коммерческих проектах и имеют большие сообщества разработчиков.

Одним из главных преимуществ Python для написания серверной части веб-приложения является его простота и быстрота разработки благодаря простоте синтаксиса и мощному набору инструментов. Кроме того, Python позволяет написать программу в относительно небольшом количестве строк кода, что делает его идеальным выбором для быстрой разработки прототипов.

Но, как и любой язык программирования, у Python есть свои ограничения. Например, он может быть не столь быстрым и неэффективным в обработке больших объемов данных, что может быть важным фактором при выборе языка для создания успешных и популярных веб-приложений.

В целом можно сказать, что Python - отличный выбор при создании серверной части веб-приложения, благодаря своей гибкости, простоте и эффективности. Однако, конечный выбор всегда зависит от особенностей конкретного проекта и условий его разработки.

* + 1. **Node.js (JavaScript)**

Node.js - это высокопроизводительная среда исполнения JavaScript, которая позволяет запускать код на стороне сервера. Node.js использует асинхронную модель программирования и событийно-ориентированную архитектуру, что позволяет обеспечить высокую производительность и масштабируемость.

Node.js отлично подходит для написания серверной части веб-приложений, так как он имеет множество возможностей, позволяющих легко работать с сетевыми запросами и базами данных. Node.js также предоставляет множество модулей для работы с различными сервисами и протоколами, такими как HTTP, TCP, UDP, WebSocket, SMTP и другие.

Благодаря тому, что Node.js использует язык JavaScript, который многие разработчики уже знают, он предоставляет простой и удобный способ для создания северной части веб-приложения. Кроме того, Node.js использует модульную структуру, что позволяет создавать гибкую и масштабируемую архитектуру приложения.

Он обеспечивает высокую производительность, простоту в использовании и множество возможностей для написания эффективного кода, но, как и любая другая технология, Node.js имеет свои недостатки, которые могут быть важными при разработке серверной части веб-приложения. Некоторые из этих недостатков перечислены ниже:

1. Сложность в работе с большими файлами: при работе с большими файлами Node.js может столкнуться с проблемами, так как технология работает с использованием памяти.

2. Медленные операции ввода/вывода: Node.js может быть медленным при обработке операций ввода/вывода, таких как чтение файлов и сетевых запросов, что может вызвать задержки для пользователя.

3. Неоднородность пакетов: Некоторые модули Node.js могут быть устаревшими или написаны плохо, что может привести к проблемам при использовании.

На практике, часть недостатков может быть минимизирована, поэтому данная технология считается хорошим выбором для создания серверной части веб-приложений.

* + 1. **C#**

C# является одним из самых популярных языков программирования на платформе .NET. Он имеет множество преимуществ, таких как:

1. Быстрота разработки: C# основан на объектно-ориентированной парадигме программирования, что значительно упрощает и ускоряет разработку.

2. Безопасность: C# применяет концепцию безопасности типов, что позволяет уменьшить количество ошибок и улучшить надежность программного кода.

3. Мощные библиотеки классов: .NET Framework содержит обширную библиотеку классов, позволяющую быстро написать программное обеспечение любой сложности.

4. Интеграция с другими языками: C# может взаимодействовать с другими языками программирования на платформе .NET, такими как Visual Basic .NET, F# и т.д.

Относительно вопроса о том, насколько C# подходит для написания серверной части веб-приложения, можно сказать, что этот язык программирования отлично подходит для этой задачи. С помощью фреймворков [ASP.NET](https://vk.com/away.php?utf=1&to=http%3A%2F%2FASP.NET) и [ASP.NET](https://vk.com/away.php?utf=1&to=http%3A%2F%2FASP.NET) Core можно написать мощные серверные приложения и веб-сервисы. Они обеспечивают высокую скорость работы, масштабируемость и безопасность. Кроме того, C# позволяет легко работать с базами данных, что является необходимым условием при разработке серверной части веб-приложения.

Несмотря на то, что C# является одним из самых подходящих языков программирования для создания серверной части веб-приложения, у него всё же есть некоторые недостатки:

1. Сложность отладки: поскольку C# - это язык со строгой типизацией, процесс отладки бывает часто затруднен. Для поиска и устранения ошибок в коде могут потребоваться значительные усилия.

2. Потребление памяти: при создании серверной части веб-приложения на C# может потребоваться больше памяти, чем на некоторых других языках.

3. Сложность обработки асинхронных запросов: в C# асинхронные операции реализуются через использование ключевых слов async и await. Это может быть сложно для начинающих разработчиков, и в некоторых случаях может привести к ошибкам в программе.

4. Ограниченная кроссплатформенность: C# является основным языком программирования для платформы .NET, которая работает только на операционных системах Windows. Хотя существует открытое программное обеспечение, которое позволяет запускать приложения на C# на других платформах, это может потребовать дополнительных усилий и времени.

Вывод:

Для написания серверной части было принято решение выбрать язык программирования Python по ряду причин:

* Простота и читаемость кода: Python имеет лаконичный и понятный синтаксис, который делает код читабельным и легко поддерживаемым.
* Опыт разработки: имеется опыт разработки приложений с помощью данного языка программирования.
* Ограниченность времени: веб-приложение нужно написать в кратчайшие сроки.
* Большое сообщество и библиотеки: Python является одним из самых популярных языков программирования и имеет большое сообщество разработчиков. Это в свою очередь позволяет использовать готовые библиотеки для решения различных задач.
* Широкие возможности интеграции: Python может легко интегрироваться с другими технологиями, такими как базы данных, фреймворки, API, что позволяет создавать мощные и расширяемые приложения.
* Низкая стоимость разработки: Python имеет низкий порог вхождения для программистов, что позволяет экономить время на обучении, а также существуют бесплатные инструменты разработки.
* Безопасность: Python использует строгую типизацию данных и встроенные механизмы безопасности, что снижает риск возможных ошибок и уязвимостей в приложении.

## Выбор технологий

**Клиентская часть.**

Bootstrap5 - это один из самых популярных и широко используемых фреймворков для фронтенд-разработки веб-приложений. Он был создан на базе HTML, CSS и JavaScript, что делает его легко доступным и понятным даже для начинающих разработчиков.

Основные причины, по которым для разработки приложения был выбран данный фреймворк:

* Bootstrap5 предлагает простую и эффективную систему сетки, которая помогает упорядочить контент и достичь согласованности дизайна на разных устройствах и платформах. Это значительно упрощает процесс разработки и оптимизации веб-страницы.
* Bootstrap5 предоставляет большой выбор компонентов и элементов дизайна для веб-приложений, таких как кнопки, формы, модальные окна, навигационные меню, карусели и т.д. Они все готовы к использованию и могут быстро интегрироваться в проект.
* Гибкость и расширяемость: он позволяет легко добавлять пользовательские стили и компоненты, что делает его идеальным инструментом для создания уникальных и индивидуальных дизайнов.

Все эти факторы делают Bootstrap5 сильной и надежной платформой для создания клиентской части веб-приложений.

jQuery - это библиотека JavaScript, которая упрощает написание кода. Она позволяет разработчикам быстро и легко манипулировать DOM (объектная модель документа), обрабатывать события, анимировать элементы, работать с AJAX и многими другими задачами. Данная библиотека может значительно упростить обработку пользовательских событий и взаимодействие с пользователями. Также jQuery обладает простым и удобным синтаксисом.

Инструмент Jinja2 был выбран для того, чтобы генерировать HTML-код на стороне сервера. Он облегчает создание динамически генерируемых страниц, и это позволяет использовать шаблоны для повторяющихся элементов сайта. Jinja2 упрощает процесс создания страниц, а также позволяет создавать легко поддерживаемый и модифицируемый код.

Сочетание jQuery и jinja2 является мощным инструментом для создания современных и динамичных веб-приложений.

**Серверная часть.**

Фреймворк Flask - это миниатюрный и легковесный фреймворк для создания серверной части веб-приложений на языке программирования Python.

Технология была выбрана Flask по нескольким причинам:

* Является одним из самых популярных фреймворков для создания веб-приложений на Python, что означает, что есть множество библиотек, документации и сообщества, которые могут помочь в разработке.
* Обладает простотой и легковесностью, что позволяет быстро разработать MVP (Минимально жизнеспособный продукт) и затем постепенно добавлять новые функции.
* Является гибким фреймворком, который позволяет разработчикам выбирать наиболее подходящие инструменты и библиотеки для своих целей.
* Используя Flask, разработчики могут легко интегрировать другие библиотеки Python, такие как SQLAlchemy для работы с базами данных и Jinja2 для создания HTML-шаблонов.

Библиотека os (от англ. operating system) - это стандартная библиотека языка программирования Python, которая предоставляет функции для работы с операционной системой, такие как доступ к файловой системе, обращение к переменным окружения, управление процессами, создание каталогов, удаление файлов и другое.

Библиотека os была выбрана, потому что она предоставляет удобные средства для работы с файловой системой и операционной системой в целом. Она даёт возможность создавать, открывать и закрывать файлы, изменять их содержимое, копировать и перемещать их в нужные каталоги, а также получать информацию о файлах, например, размер, время создания, и многое другое.

## Выбор базы данных

СУБД SQLite — это легкий и быстрый в использовании относительно других СУБД, который является свободным и предоставляет кроссплатформенную поддержку. Она использует файловую систему для хранения данных, в отличие от традиционных СУБД, в которых данные хранятся в отдельном сервере. Одним из основных преимуществ SQLite является его малый размер, что делает его легче и быстрее в установке и настройке.

Данная СУБД была выбрана в связи с тем, что она позволяет хранить данные локально на сервере без необходимости каскадной операции сетью. Это позволяет уменьшить нагрузку на сервер, ускорить работу приложения и упростить его настройку и установку. Кроме того, SQLite обладает прекрасной производительностью, что позволяет быстро получать и обрабатывать данные.

SQLAchemy - это мощный инструмент, который позволяет нам работать с базами данных в приложениях, написанных на Python. Он облегчает работу с базами данных, а также позволяет использовать ORM (объектно-реляционное отображение) для работы с данными в базе.

ORM - это технология, которая позволяет нам работать с базой данных в терминах объектов и классов в Python. Благодаря этому мы можем использовать привычный синтаксис Python для работы с базой, вместо того чтобы использовать SQL язык.

Использование SQLAlchemy и ORM позволяет легко и быстро работать с базой данных. Они упрощают изменение данных, запросы и поиск информации в базе данных. Так же это делает код чище и более легким в понимании, который можно легко поддерживать и расширять в будущем.

## Выбор среды разработки

Выбор среды разработки очень важен для программиста, поскольку работа с ней занимает значительную часть времени. Он во многом зависит от языка программирования, который используется в проекте. Каждая среда обладает определенными функциональными возможностями для работы с определенным языком программирования и набором библиотек. Некоторые среды разработки имеют более простой и интуитивный интерфейс, другие - более продвинутые функциональности. Среда разработки облегчает процесс написания кода, обеспечивает средства отладки и тестирования, предоставляет инструменты для управления версиями и обмена кодом между участниками команды.

**Клиентская часть.**

Для клиентской части был выбран редактор кода Visual Studio Code. Это кроссплатформенный бесплатный редактор кода, созданный компанией Microsoft. Он поддерживает множество языков программирования, включая HTML, CSS, JavaScript, Python и многие другие.

Вот несколько причин, почему Visual Studio Code стоит выбрать для написания клиентской части веб-приложения:

1. Интегрированная среда разработки: Visual Studio Code включает в себя множество полезных функций, таких как автодополнение кода, форматирование кода, отладка и управление версиями. Вам не придется использовать дополнительные инструменты и расширения, чтобы выполнить большую часть своей работы.

2. Множество плагинов и расширений: Visual Studio Code имеет широкую поддержку и активное сообщество, позволяющее использовать различные плагины и расширения для улучшения работы с редактором кода. Вы можете улучшить свой опыт разработки, устанавливая и настраивая различные плагины.

3. Хорошая производительность: Visual Studio Code известен своей быстрой загрузкой времени и выполнением большинства операций в режиме реального времени. Это делает его идеальным инструментом для разработки веб-приложений.

Кратко о функциях VS Code:

* Мощный редактор кода с поддержкой синтаксиса для многих языков программирования;
* Интеграция с Git;
* Отладчик кода;
* Расширяемость и гибкие настройки;
* Широкая поддержка фреймворков и языков программирования;
* Бесплатный и с открытым исходным кодом.

**Серверная часть.**

Для написания серверной части приложения была выбрана интегрированная среда разработки для языка программирования Python – PyCharm. Она представляет собой мощный и удобный инструмент, который позволяет быстро и просто создавать, отлаживать и разрабатывать приложения.

Что касается разработки серверной части веб-приложений, PyCharm предоставляет ряд преимуществ:

1. Поддержка различных фреймворков: PyCharm поддерживает все популярные фреймворки Python для веб-разработки, включая Flask, который будет использован в проекте.

2. Инспектирование кода: PyCharm предоставляет подсказки и автодополнения кода, а также позволяет быстро переходить к определению функций и классов.

3. Отладка: в PyCharm есть удобный и мощный отладчик, который позволяет отслеживать исполнение кода и находить ошибки.

4. Управление проектом: PyCharm предоставляет множество инструментов для управления проектом, включая возможность автоматической установки зависимостей, Git-интеграцию и многое другое.

6. Редактирование файлов – PyCharm предоставляет удобный редактор, который позволяет быстро и удобно редактировать файлы Python, HTML, CSS и JavaScript.

Добавить сравнение с аналогами (Moodle, Google-таблицы)

# РАЗРАБОТКА ПРИЛОЖЕНИЯ

## Архитектурная часть приложения

Общий рисунок архитектуры

Дерево файлов проекта на стороне клиента

Дерево файлов проекта на стороне сервера

## Информационная часть приложения

Информационная часть реализована с помощью программной библиотеки SQLAlchemy. Технология ORM позволяет нам работать с базой данных в терминах объектов и классов в Python.

В таблице 1 представлено краткое описание всех таблиц приложения.

**Таблица 1 – Краткое описание таблиц базы данных**

|  |  |
| --- | --- |
| Название таблицы | Краткое описание |
| User | Реализует хранение данных о пользователе (преподавателе или студенте) и имеет связь с таблицами: Group через поле group\_id и PointData через поле point\_data\_list. |
| Subject | Реализует хранение данных о предмете (дисциплине). Содержит связь с таблицей Statement через поле statements. |
| Group | Реализует хранение данных о группе. Содержит связь с таблицей User через поле students. |
| Statement | Реализует хранение данных о ведомости. Содержит связи с таблицей Subject через поле subject\_id, с таблицей Group через поле groups и с таблицей Point через поле points. |
| Point | Реализует хранение данных о столбике с работами. Содержит связи с таблицей PointData через поле point\_data\_list и с таблицей Statement через поле statement\_id. |
| PointData | Реализует хранение данных об отправленных работах. Содержит связи с таблицей Point через поле point\_id, с таблицей User через поле user\_id и с таблицей Remark через поле remarks. |
| Remark | Реализует хранение данных о комментариях, оставленных преподавателем. Содержит связь с таблицей PointData через поле point\_data\_id. |

Расписать, как поля выглядят в БД

Привести скриншоты содержания полей БД в СУБД

Так как СУБД при своем первом старте имеет только одну базу данных по умолчанию, которая не имеет структуру, то первым делом до запуска серверной части приложения, нужно создать таблицы, связи между ними, индексы для полей, по которым часто выполняется оператор SELECT. Для автоматизации создания схемы базы данных используются заранее написанные SQL скрипты, называемые миграциями [8]. Одна часть из них, помеченная как 16 «up», выполняет создание новых таблиц, связей, индексов и типов. Вторая часть, помеченная как «down», откатывает примененные изменения. Для того чтобы не запускать скрипты самостоятельно мы используем утилиту[9], для которой есть поддержка баз данных, которыми управляет Postgres SQL [10]. Так же для того чтобы упростить загрузку миграций в контейнерной среде Docker, была написана утилита [11], которая через volume функционал монтирует файлы миграции и запускает их.

## Клиентская часть приложения

Сюда добавить коды:

- форма ведомости (включая):

- добавления файла работы и отчёта (показать более подробно)

- поле для проставления оценки

- поле для комментария

Все, что ниже – в тестирование приложения

**Авторизация и регистрация.**

Для работы в приложении пользователь должен пройти процесс регистрации или авторизации. На рисунках 3, 4 представлены окна авторизации и регистрации пользователя.

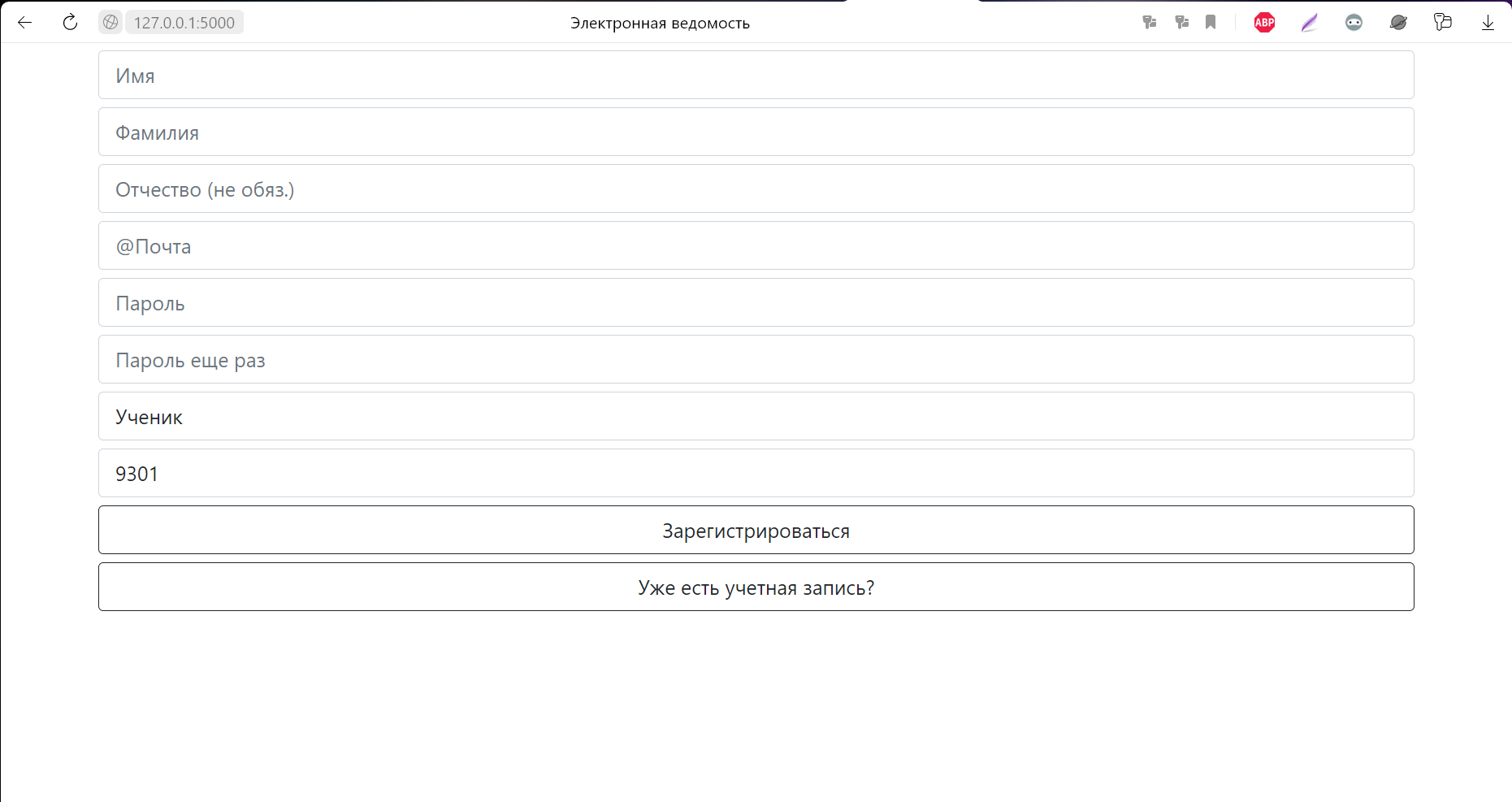


Рисунок 1 – окно регистрации пользователя

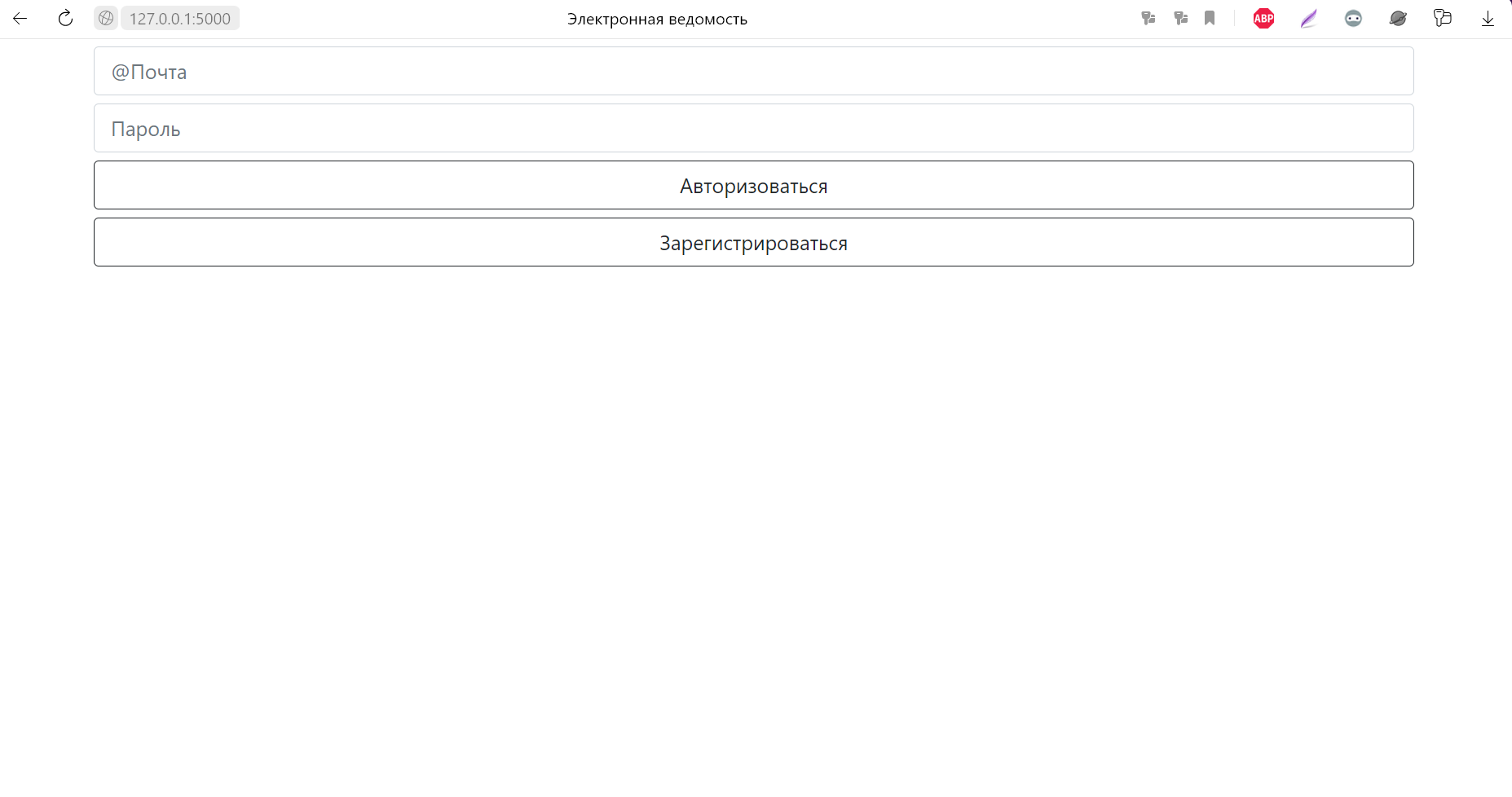


Рисунок 2 – окно авторизации пользователя

**Основной интерфейс приложения.**

В приложении предусмотрено два типа работы в зависимости от роли пользователя:

* Студент;
* Преподаватель.

При входе в аккаунт у преподавателя открывается экран, на котором отображены все существующие ведомости, а также доступно три кнопки:

* «Главная»;
* «Добавить ведомость»;
* «Выйти из учетной записи».

На рисунке 3 представлен экран преподавателя после входа в аккаунт.

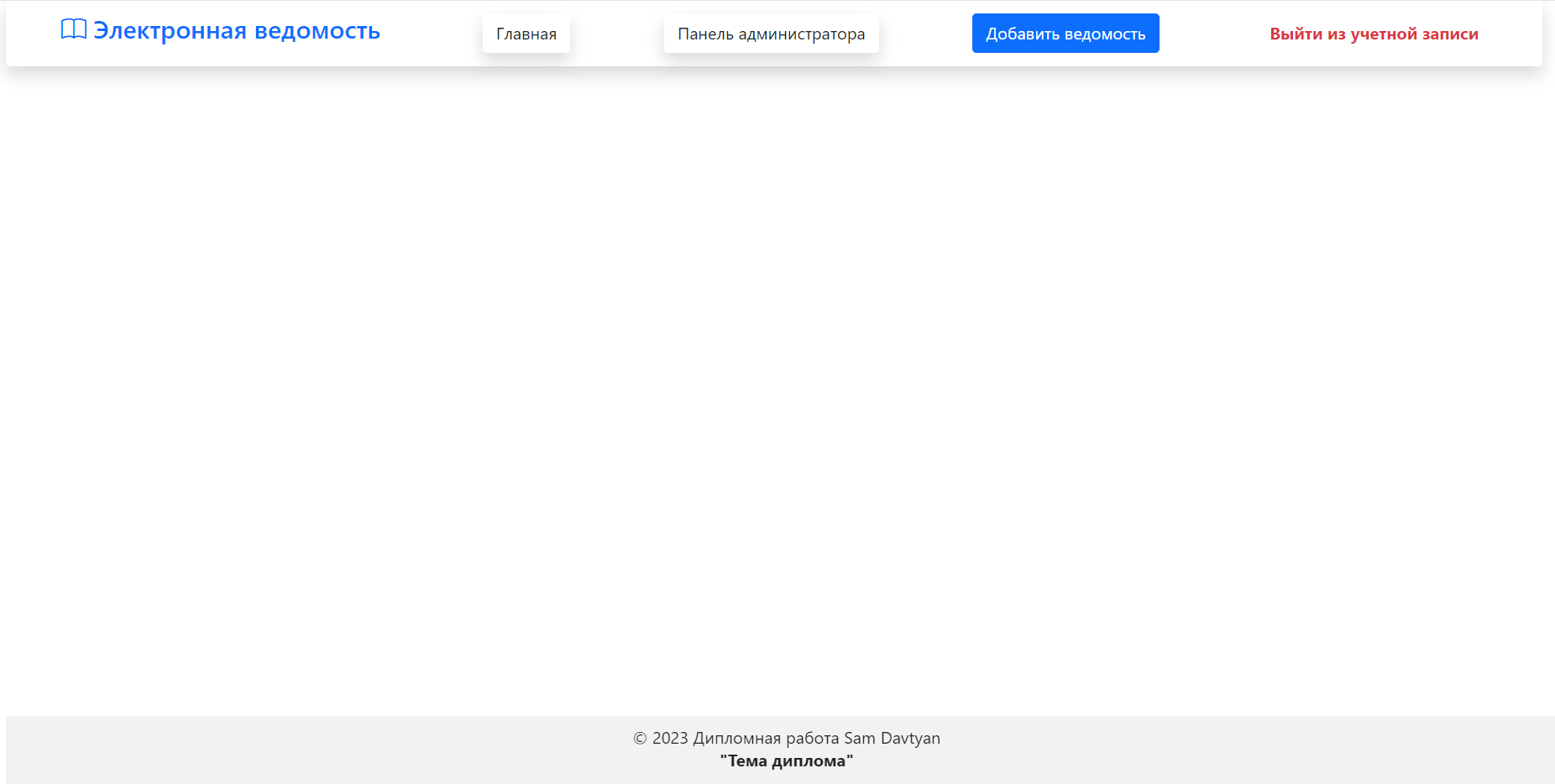


Рисунок 3 – экран преподавателя после входа в аккаунт

Для студента также открывается экран с доступными ему ведомостями и две кнопки:

* «Главная»;
* «Выйти из учетной записи».

На рисунке 4 представлен экран студента после входа в аккаунт.

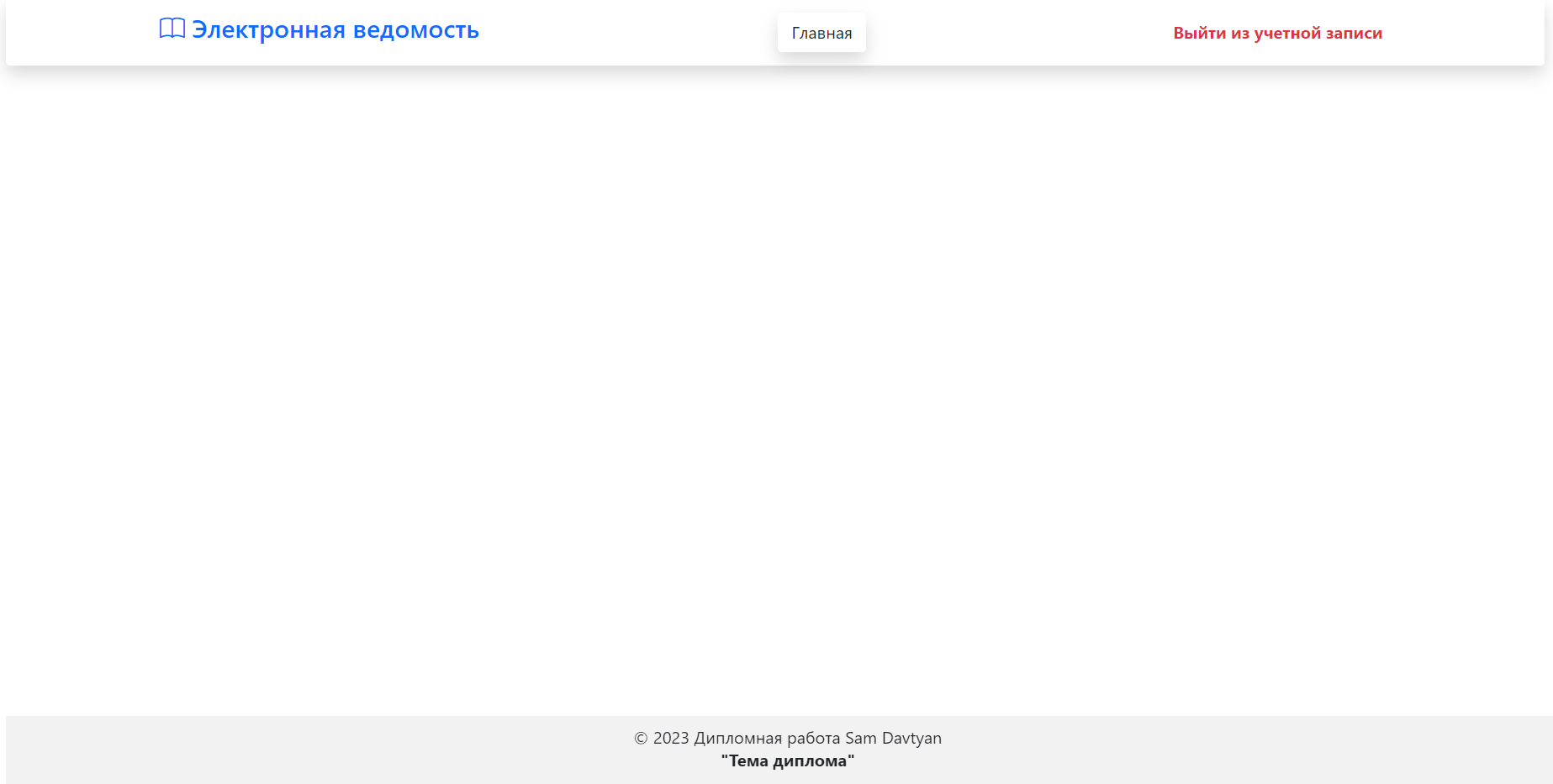


Рисунок 4 – экран преподавателя входа в аккаунт

Далее для преподавателя доступна возможность добавления ведомости. При нажатии на кнопку «Добавить ведомость» появляется всплывающее окно, в котором преподаватель может указать:

* «Семестр» – текстовое поле, в которое преподаватель может вписать номер текущего семестра, по которому создается ведомость.
* «Год» – поле с календарем, в котором преподаватель может выбрать текущий учебный год.
* «Дисциплина» – выпадающее окно, в котором преподаватель может выбрать дисциплину, по которой будет создана ведомость. Дисциплины заранее должны быть добавлены администратором.

На рисунке 5 представлено всплывающее окно для преподавателя, с помощью которого создается ведомость.

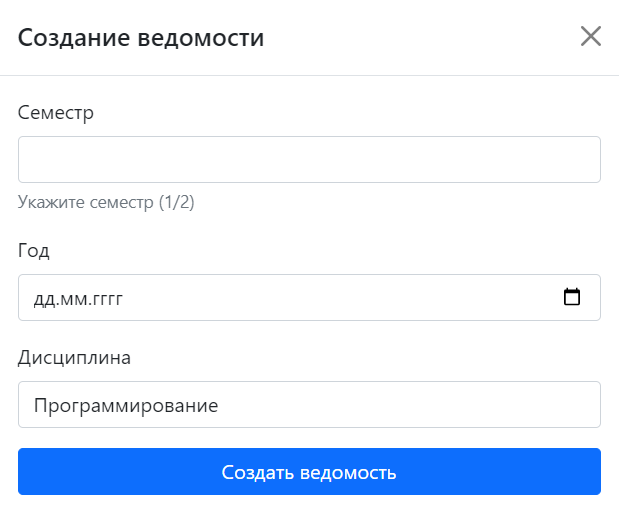


Рисунок 5 – всплывающее окно создания ведомости

После нажатия кнопки «Создать ведомость» на экране преподавателя и студентам появляется созданная ведомость и запись о том, что она успешно добавлена.

На рисунке 6 представлен экран преподавателя после создания ведомости.

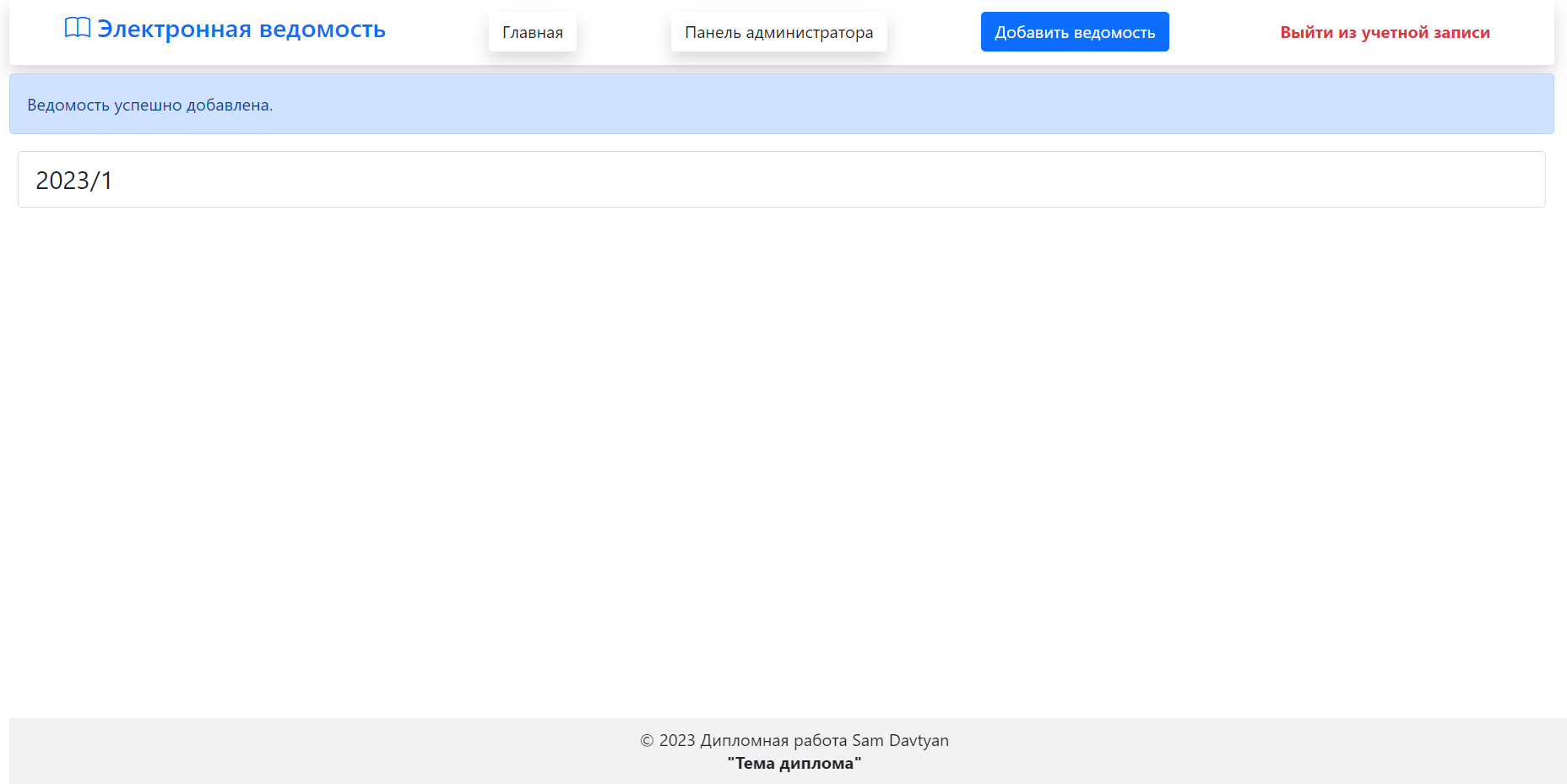


Рисунок 6 – Экран преподавателя после добавления новой ведомости

После нажатия на ведомость, преподаватель попадает на экран ведомости. Изначально он состоит из страницы, на которой указано:

* название дисциплины;
* список добавленных групп;
* пустая ведомость.

На рисунке 7 представлен экран преподавателя после перехода в только что созданную ведомость.

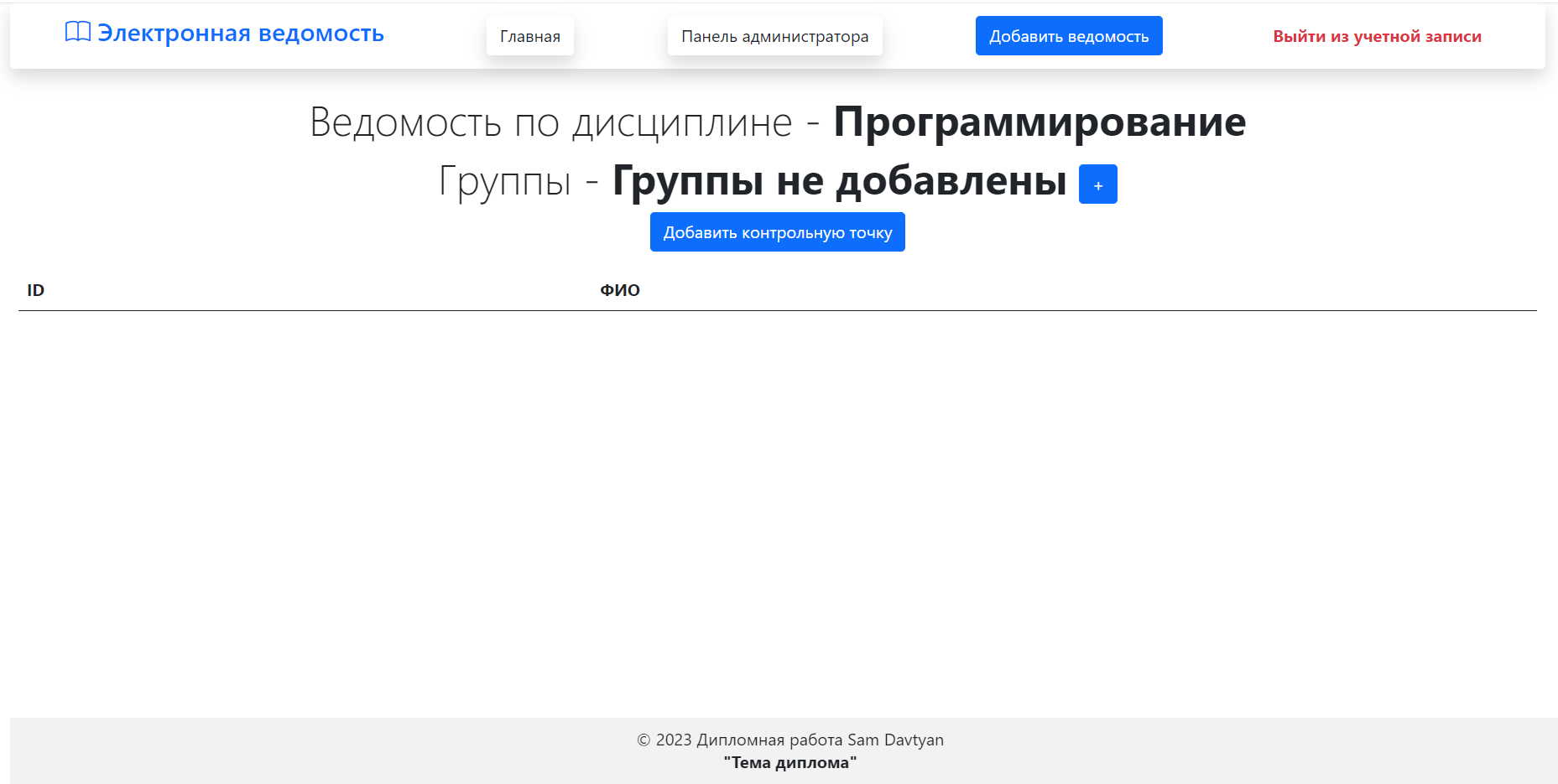


Рисунок 7 – Страница только что созданной ведомости

Преподаватель может добавить группы и контрольные точки. Для добавления группы он должен нажать на кнопку «+». При нажатии данной кнопки появляется всплывающее окно, с помощью которого преподаватель может добавить группу. Всплывающее окно состоит из одного выпадающего списка, в котором преподаватель выбирает номер группы. Номера доступных групп добавляются администратором.

На рисунках 8, 9 представлено всплывающее окно добавления группы.

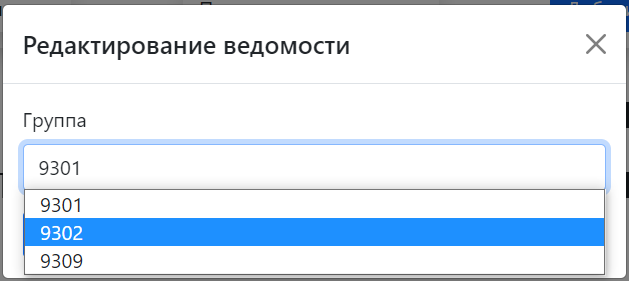


Рисунок 8 – выпадающий список для выбора номера группы

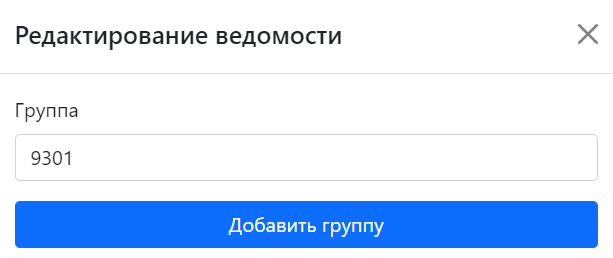


Рисунок 9 – всплывающее окно для добавления группы

После того, как преподаватель добавляет группу, в ведомости появляются все студенты, обучающиеся в данной группе.

На рисунке 10 представлена ведомость после добавления в нее группы 9302. Видно, что обучающиеся в группе 9302 и имеющие аккаунт в данном приложении студенты появились в ведомости.

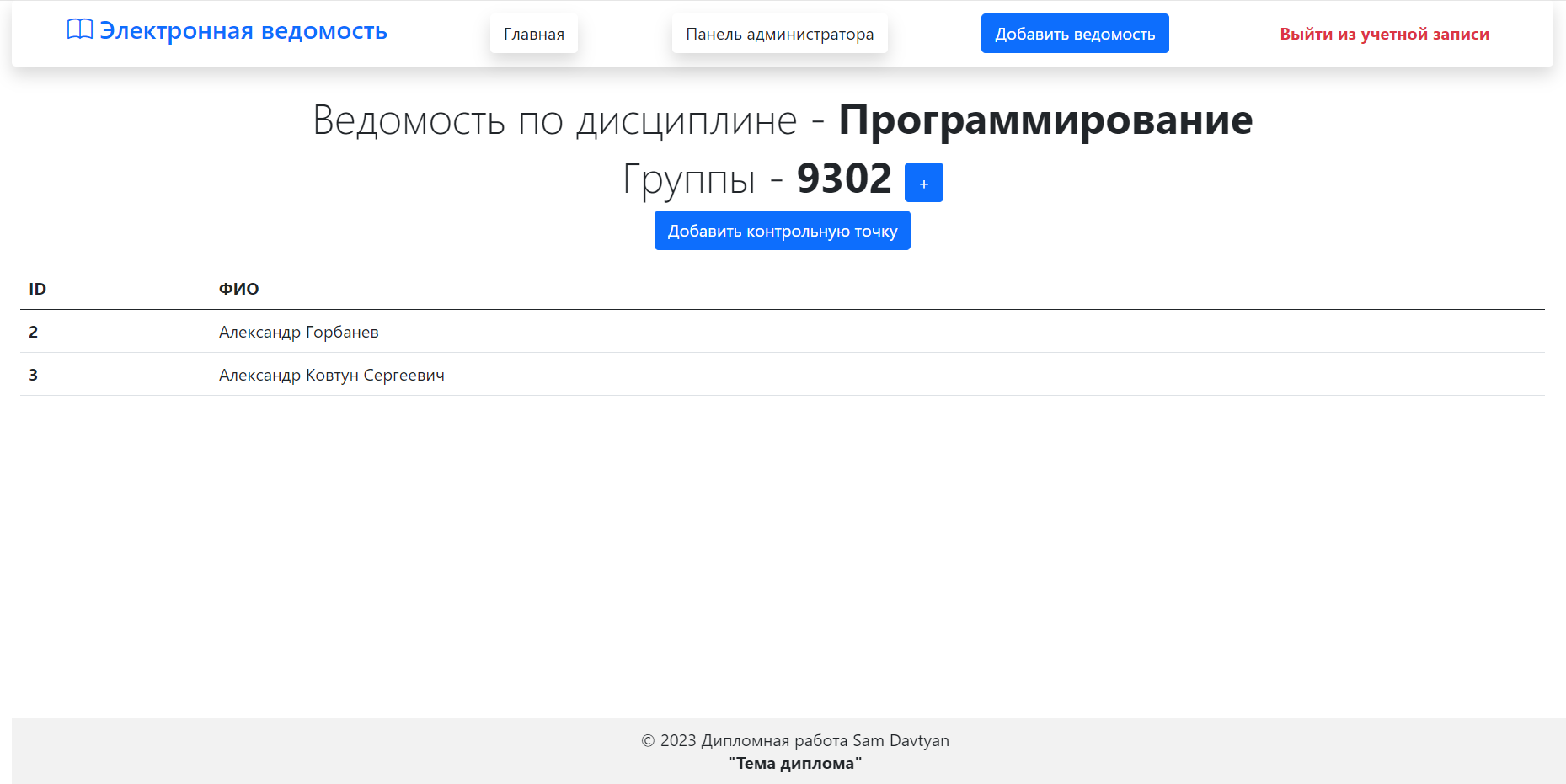


Рисунок 10 – ведомость после добавления группы

Далее преподаватель может добавить контрольные точки. Для этого он может нажать на кнопку «Добавить контрольную точку». После этого появится всплывающее окно, в котором преподаватель в текстовом поле может вписать название контрольной точки, которая в виде столбика появится в ведомости.

На рисунке 10 представлено всплывающее окно для добавления новой контрольной точки.

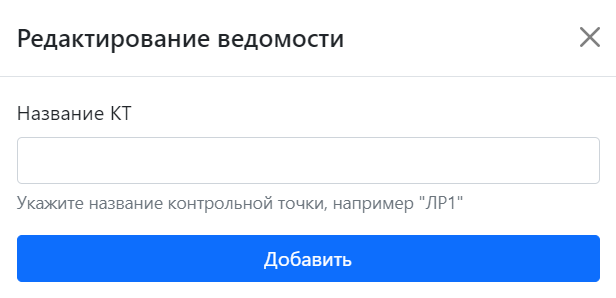


Рисунок 11 – всплывающее окно добавления контрольной точки

После добавления контрольной точки, она появляется в виде столбика таблицы. После добавления нескольких контрольных точек у ведомости появляется вид таблицы.

На рисунках 12, 13 представлена ведомость после добавления контрольных точек.

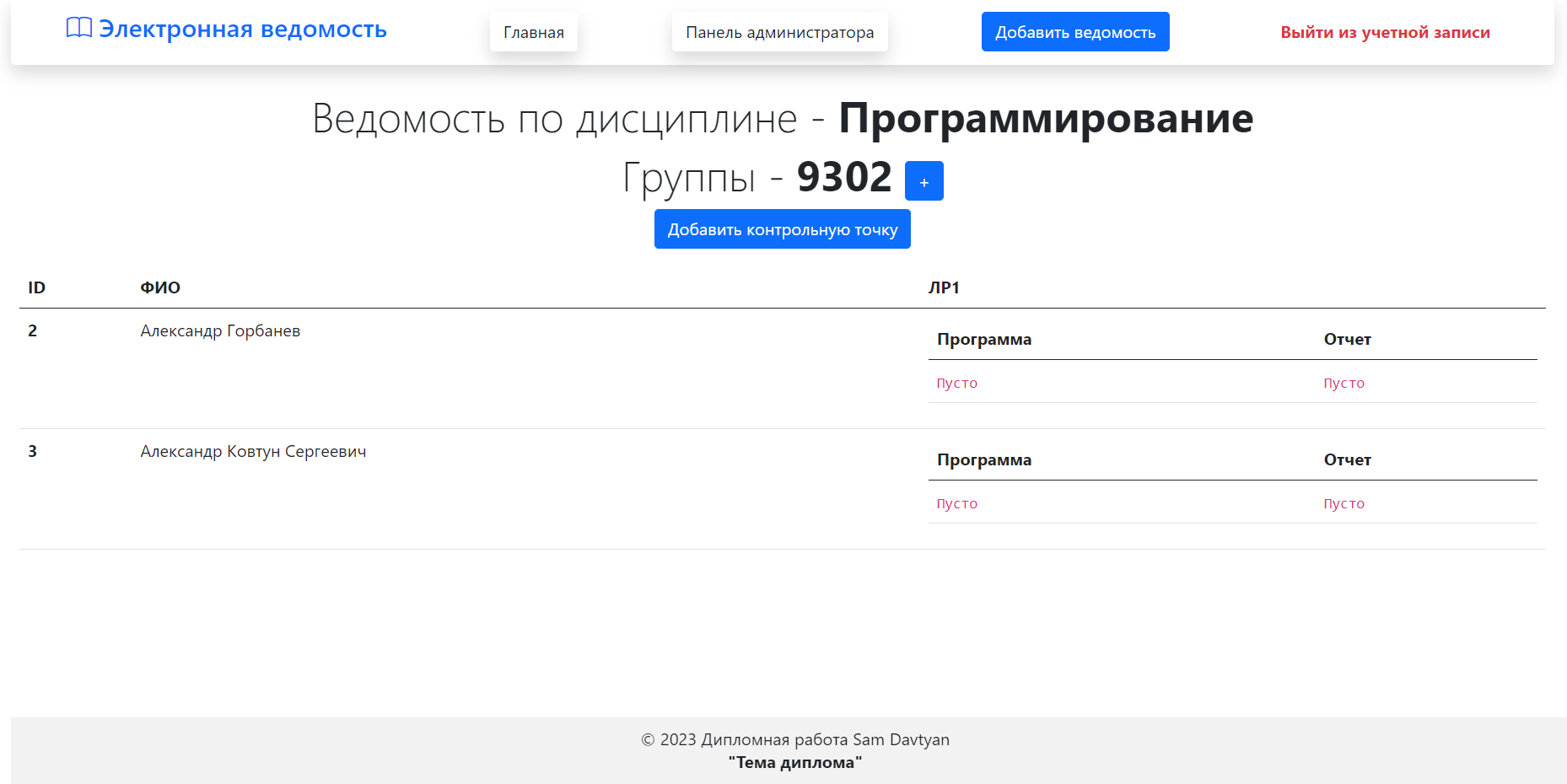


Рисунок 12 – ведомость после добавления первого столбца

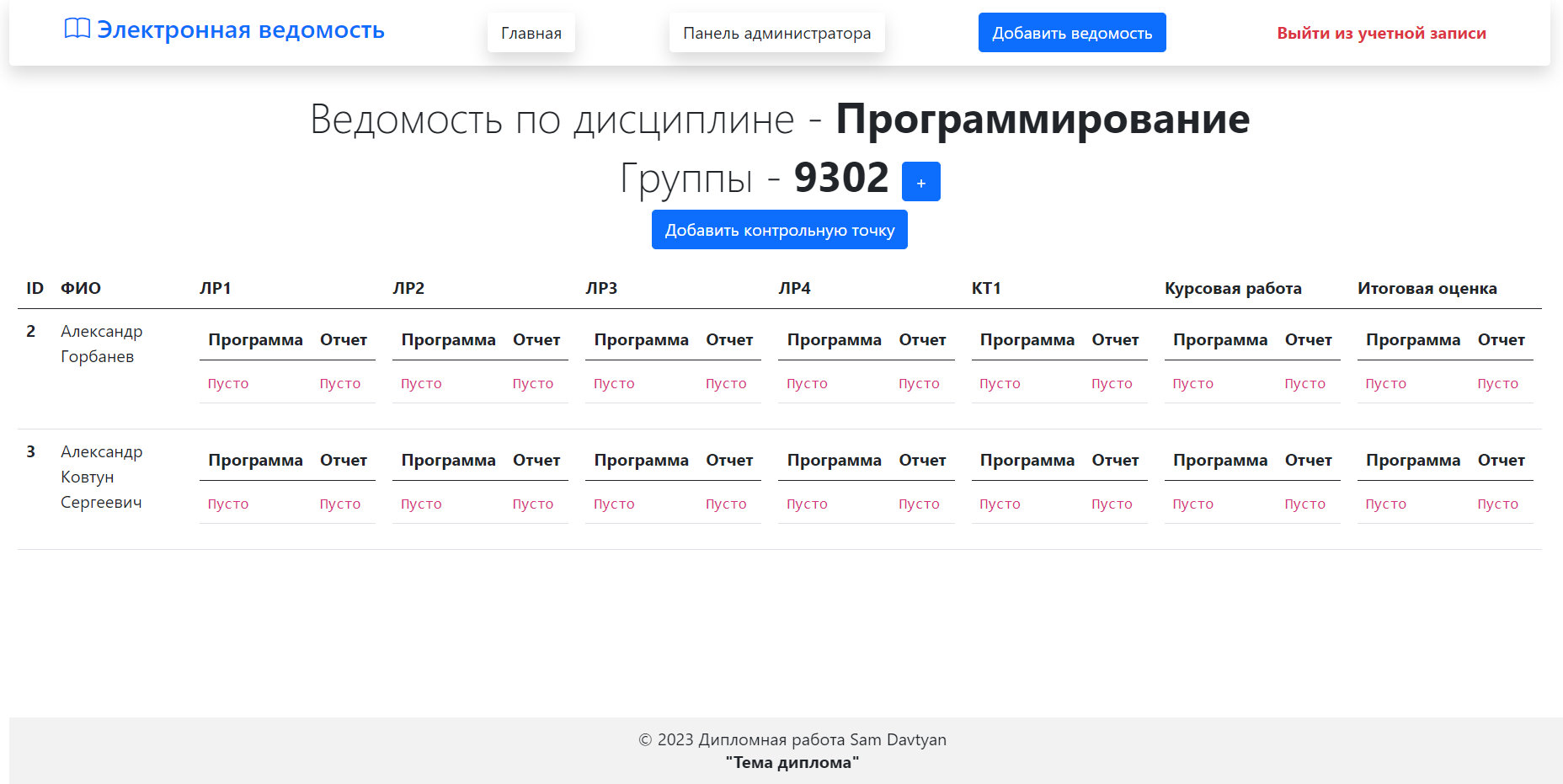


Рисунок 13 – ведомость после добавления всех столбцов

В каждом столбце присутствуют две ячейки: «Программа» и «Отчет». Изначально они пустые, потому что пока что студенты ничего не добавляли.

Ведомость появляется на странице у студентов только после того, как преподаватель добавит их группу с помощью кнопки «+».

На рисунке 14 представлена страница, которую видят студенты после того, как преподаватель предоставил им доступ к ведомости.

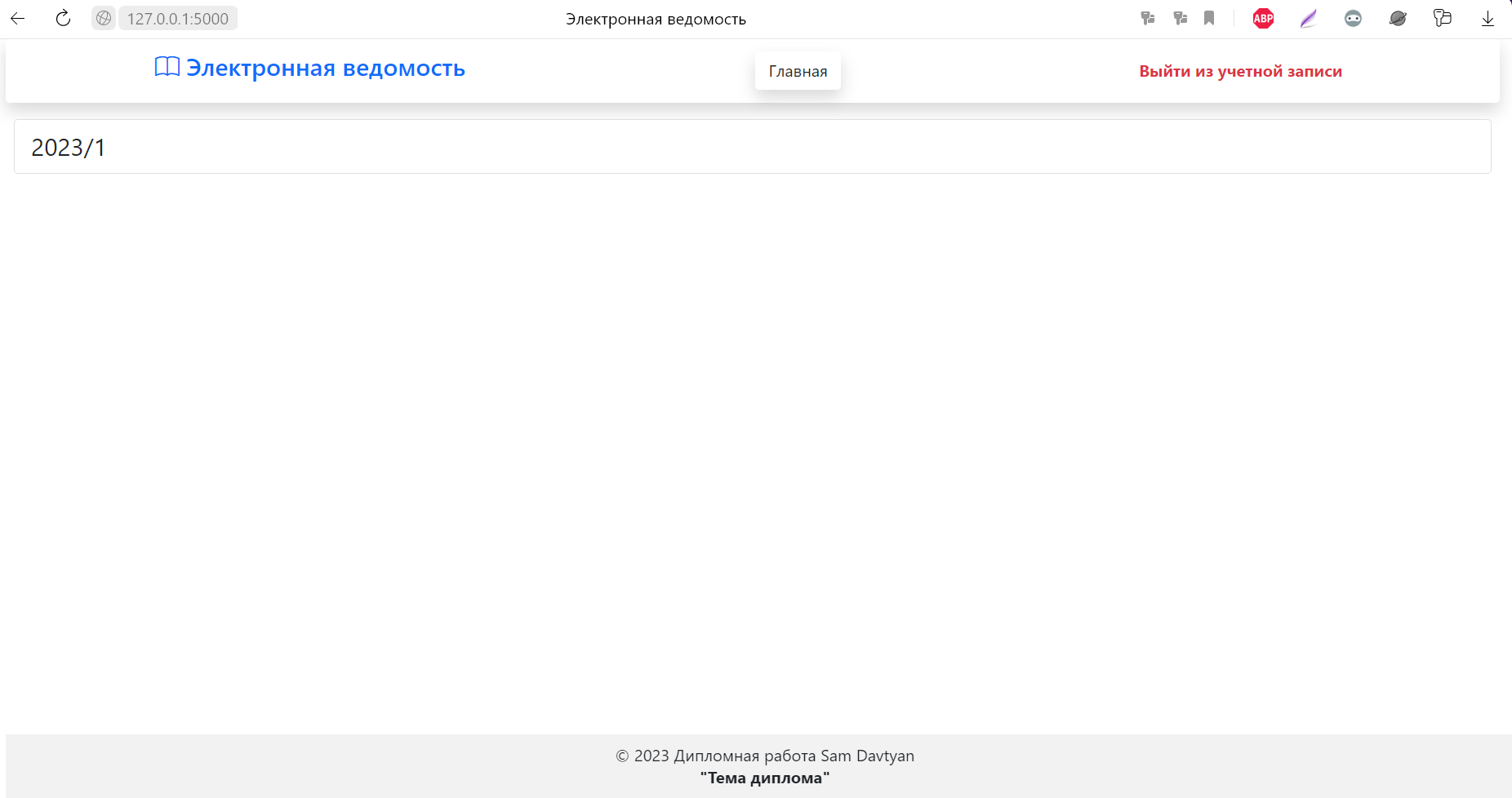


Рисунок 14 – экран студентов после предоставления им доступа к ведомости

После нажатия на ведомость, студент переходит на страницу с ведомостью, на которой указано:

* название дисциплины;
* номера групп, которым предоставлен доступ;
* ведомость в виде таблицы.

Авторизованный пользователь может добавить файлы программы и отчета к каждой контрольной точке в своей строчке. К строчкам, относящимся к другим студентам, доступа сейчас нет, в дальнейшем студенты смогут просматривать оценки друг друга.

На рисунке 15 представлена страница ведомости, которую видит студент.

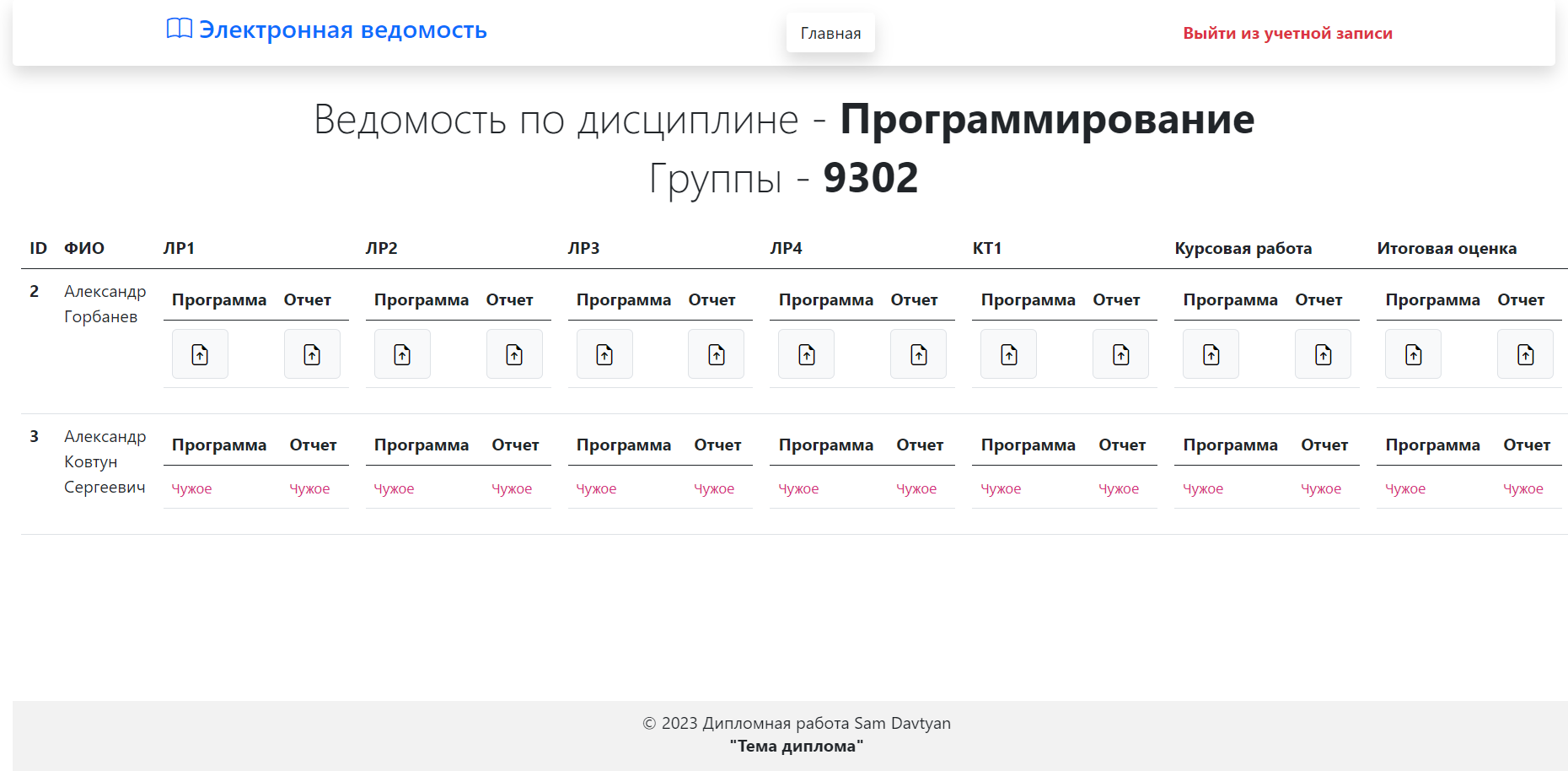


Рисунок 15 – страница ведомости для авторизованного пользователя

Для студентов под каждой ячейкой с программой и отчетом появляется иконка, нажав на которую, они могут загрузить файлы.

На рисунке 16 представлено окно, которое открывается после нажатия на кнопку.

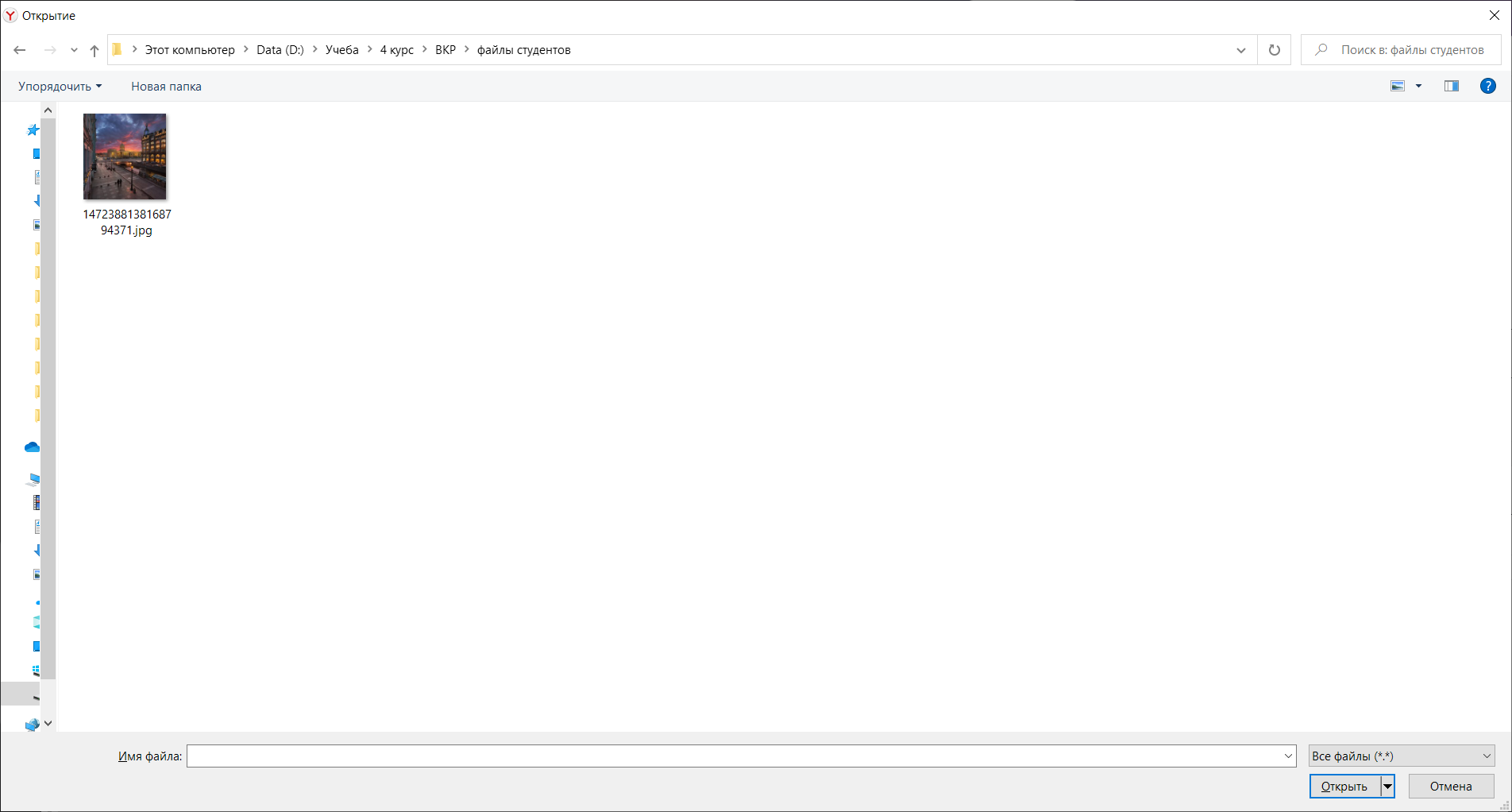


Рисунок 16 – окно с загрузкой файлов

После загрузки файла иконка подсвечивается желтым цветом, чтобы видно было, что работа ожидает проверки. В дальнейшем будет реализовано, что при проверке преподавателя и проставлении оценки, иконка будет подсвечиваться зеленым цветом, а при отправлении на доработку будет подсвечено красным.

На рисунке 17 представлена страница с ведомостью после загрузки файла студентом.

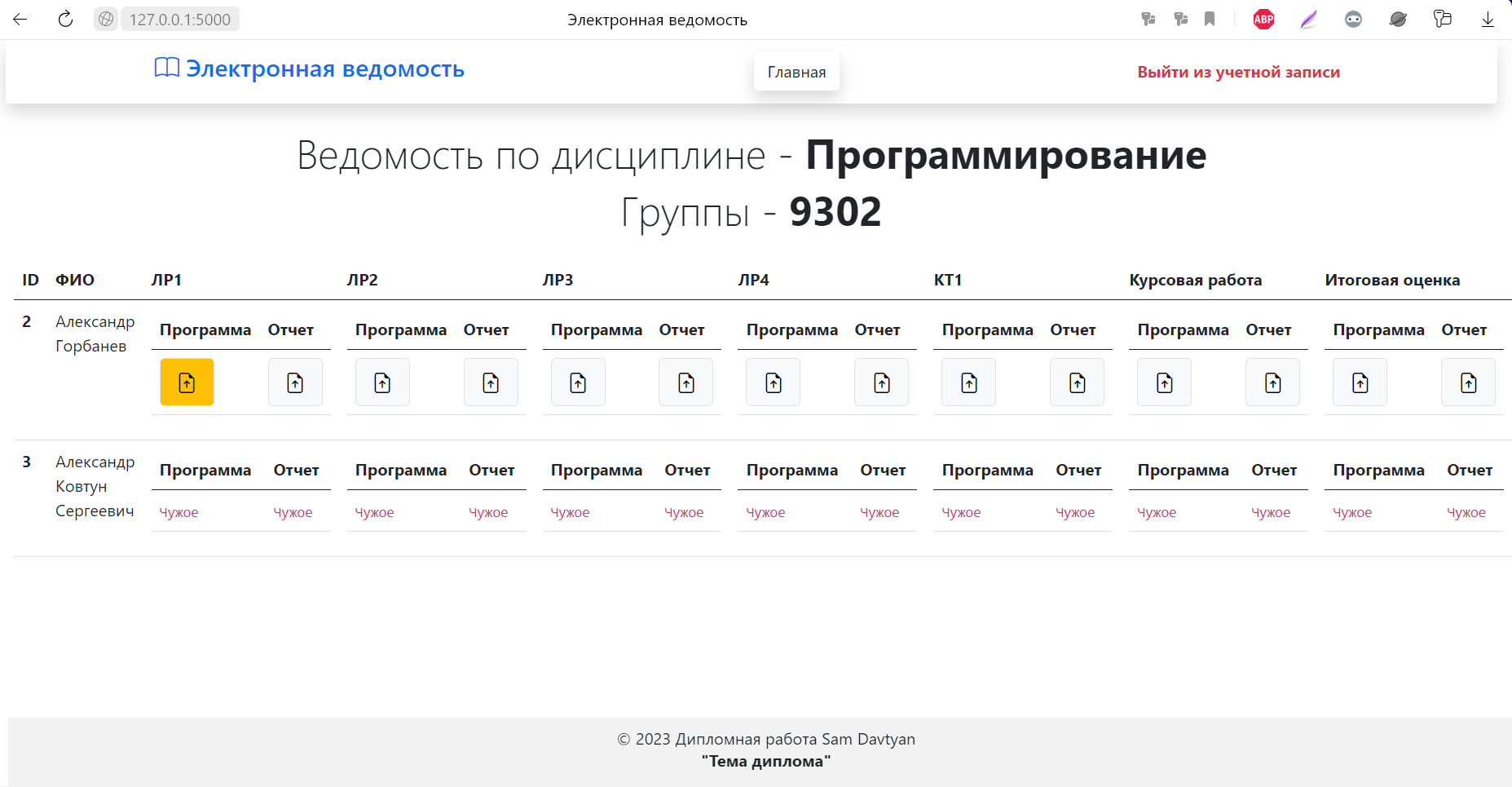


Рисунок 17 – страница ведомости после загрузки файла

После этого у преподавателя высветится, что данный студент загрузил работу. Также под иконкой загрузки файла появятся выпадающий список с возможными оценками и кнопка, нажав на которую, преподаватель может оставить комментарий. На рисунках 18, 19, 20 представлены выпадающий список с возможными оценками, кнопка для оставления комментария и сама страница, которую видит преподаватель после загрузки файла студентом.

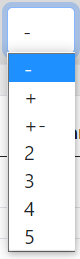


Рисунок 18 – выпадающий список с возможными оценками



Рисунок 19 – кнопка для оставления замечания

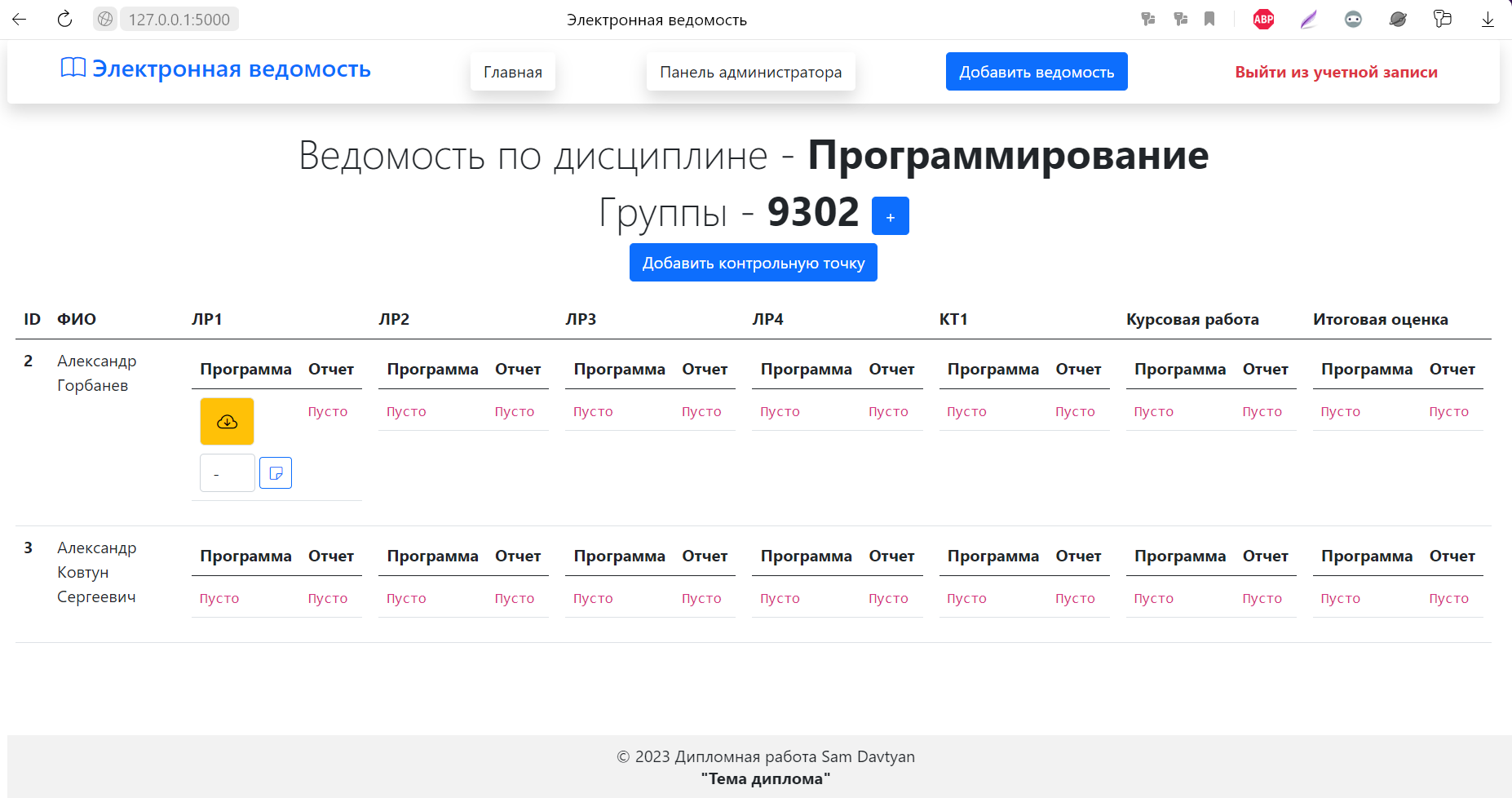


Рисунок 20 – страница ведомости у преподавателя после загрузки файла студентом

## Серверная часть приложения

Коды модели данных

Коды ответа сервера на данные операции:

- запрос данных ведомости (включая):

- добавление файла работы

- проставление оценки

- добавление комментария

## Тестирование приложения

ЗАКЛЮЧЕНИЕ

В процессе выполнения выпускной квалификационной работы было создано веб-приложение для ведения ведомостей текущего контроля.

Главное ее отличие от аналогов состоит в интуитивно понятном интерфейсе, где в одном месте собрано как загрузка файлов студентами, так и обратная связь от преподавателя.

Данная разработка может также использоваться в других сферах, где нужно вести какой-либо контроль за деятельностью, связанной с загрузкой файлов, учетом качества работы и проставлением оценок.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Система управления базами данных // Википедия. URL: <https://ru.wikipedia.org/wiki/Система_управления_базами_данных> (дата обращения: 13.05.2023).
2. JavaScript // Википедия. URL: <https://ru.wikipedia.org/wiki/JavaScript> (дата обращения: 13.05.2023 г.).
3. HTML // Википедия. URL: <https://ru.wikipedia.org/wiki/HTML> (дата обращения: 13.05.2023 г.).
4. CSS // Википедия. URL: <https://ru.wikipedia.org/wiki/CSS>(дата обращения: 13.05.2023 г.).

Плюс Moodle ЛЭТИ и общий Moodle

Google таблицы

Приложение 1. Листинги файлов клиентской части приложения

Приложение 2. Листинги файлов серверной части приложения

Сюда добавить полные коды файлов, фрагменты кода которых рассматривались в отчёте ранее. Перед листингами файлов обязательно заголовок (жирным шрифтом) с названием файла.